BigDawsSSH - Project Proposal

Kenta Yoshii

April 19, 2023

1 Overview

For the final project of CS1515, I will attempt to implement **RFC-compliant Secure Shell Procotol (SSH)**. This will largely consists of three parts: **transport protocol**, **authentication protocol**, and **connection protocol**.

1.1 Protocols

1. **Transport Protocol**

   This protocol will provide a secure and confidential channel over an insecure network. Server host authentication, key exchange, encryption, and integrity protection are what mainly gets done in this part of the protocol. After this, a unique session id will be generated to be used in later protocols.

2. **Authentication Protocol**

   The authentication of client user to the server is done in this part of the protocol. The generated session id will be used here. The assumption is that it already has a authenticated server machine and an established, ecrypted communciation channel

3. **Connection Protocol**

   This protocol specifies a mechanism to multiplex multiple streams of data over the confidential and authenticated tranport.

1.2 Security

1. Confidentiality

   To assure confidentiality, I am going to use AES with CBC mode, which is a widely accepted mode of encryption in the group. To mitigate the risk of an attacker guessing the Initializatio Vector, I will be taking advantage of the **SSH_MSG_IGNORE**

2. Data Integrity

   We will use MAC to guarantee data integrity of packets being sent over the internet. We will also be using rekeying technique every 1GB to prevent from any information being gained by an attcker.

3. Man in the Middle

   Since the server host key is known to the client a priori and we use MAC scheme for data integrity, there is no risk of MitM attack.

4. User Authenticity

   To assure that we are interacting with a valid client host, we will be using Public Key Authentication. An alternative to this is using Password Authentication, but this is vulnerable to server's weak security.

2 Roadmap

Here is the roadmap that resembles the hierarchical order of protocols, which is unsurprising.

1. Implement Transport Protcol
   - (a) TCP connection establishement
   - (b) Handshake
   - (c) Algorithm Negotiations and Key Exchange
   - (d) Service Request(ssh-userauth, ssh-connection)
   - (e) (Stretch) Key Re-Exchange
   - (f) (Stretch) Support for more than one Cryptographic Algorithm for encryption
2. Implement Client Authentication Protocol
   - (a) Public Key based authentication of the client
   - (b) (Stretch) Password based authentication of the client
3. (Stretch) Implement Connection Protocol
   - (a) Opening/Closing Channels
   - (b) TCP/IP Forwarding

3 Libraries / Language / Reference

Language Golang

Library net/http, (more to be added as I see fit)

Reference RFC4250, RFC4251, RFC4252(authentication), RFC4253(transport), RFC4254(connection)